CSP Exercise 04 Solution

Eugen Sawin

June 4, 2012

Exercise 4.1

(a) The zebra problem is the constraint network $N = \langle V, D, C \rangle$ with

$$
\begin{aligned}
V =& (english, spanish, ukrainian, norwegian, japanese,\\
& red, green, ivory, yellow, blue\\
& dog, snail, fox, horse, zebra\\
& coffee, tea, milk, juice, water\\
& oldgold, kools, chesterfield, lucky, parliament)\\
D =& (D_v)_{v \in V} \text{ where } D_v = \begin{cases} \{1\}, & \text{if } v = norwegian \\ \{3\}, & \text{if } v = milk \\ \{1,2,3,4,5\}, & \text{otherwise} \end{cases}\\
R_{eq} =& \{(a,a) \mid a \in \mathbb{N}\}\\
R_{next} =& \{(a,b) \mid a,b \in \mathbb{N}, |a-b| = 1\}\\
R_{rightof} =& \{(a+1,a) \mid a \in \mathbb{N}\}\\
C =& \{R_{v_1,v_2 \in \{english,spanish,ukrainian,norwegian,japanese\}} = (D_{v_1} \times D_{v_j}) \setminus R_{eq},\\
& R_{v_1,v_2 \in \{red,green,ivory,yellow,blue\}} = (D_{v_1} \times D_{v_j}) \setminus R_{eq},\\
& R_{v_1,v_2 \in \{dog,snail,fox,horse,zebra\}} = (D_{v_1} \times D_{v_j}) \setminus R_{eq},\\
& R_{v_1,v_2 \in \{coffee,tea,milk,juice,water\}} = (D_{v_1} \times D_{v_j}) \setminus R_{eq},\\
& R_{v_1,v_2 \in \{oldgold,kools,chesterfield,lucky,parliament\}} = (D_{v_1} \times D_{v_j}) \setminus R_{eq},\\
& R_{english,red} = (D_{english} \times D_{red}) \cap R_{eq},\\
& R_{spanish,dog} = (D_{spanish} \times D_{dog}) \cap R_{eq},\\
& R_{coffee,green} = (D_{coffee} \times D_{green}) \cap R_{eq},\\
& R_{ukrainian,tea} = (D_{ukrainian} \times D_{tea}) \cap R_{eq},\\
& R_{oldgold,snail} = (D_{oldgold} \times D_{snail}) \cap R_{eq},\\
& R_{kool,yellow} = (D_{kool} \times D_{yellow}) \cap R_{eq},\\
& R_{lucky,juice} = (D_{lucky} \times D_{juice}) \cap R_{eq},\\
& R_{japanese,parliament} = (D_{japanese} \times D_{parliament}) \cap R_{eq},\\
& R_{chesterfield,fox} = (D_{chesterfield} \times D_{fox}) \cap R_{next},\\
& R_{yellow,horse} = (D_{yellow} \times D_{horse}) \cap R_{next},\\
& R_{norwegian,blue} = (D_{norwegian} \times D_{blue}) \cap R_{next},\\
& R_{green,ivory} = (D_{green} \times D_{ivory}) \cap R_{rightof}\}
\end{aligned}
$$

To answer the questions, the Norwegian drinks water and the Japanese owns the zebra. I have added the corresponding problem file in XCSP2.1 format to my project repository *(xcsp2_examples/zebra.xml)*.

(b) See figure 1 (names are abbreviated).

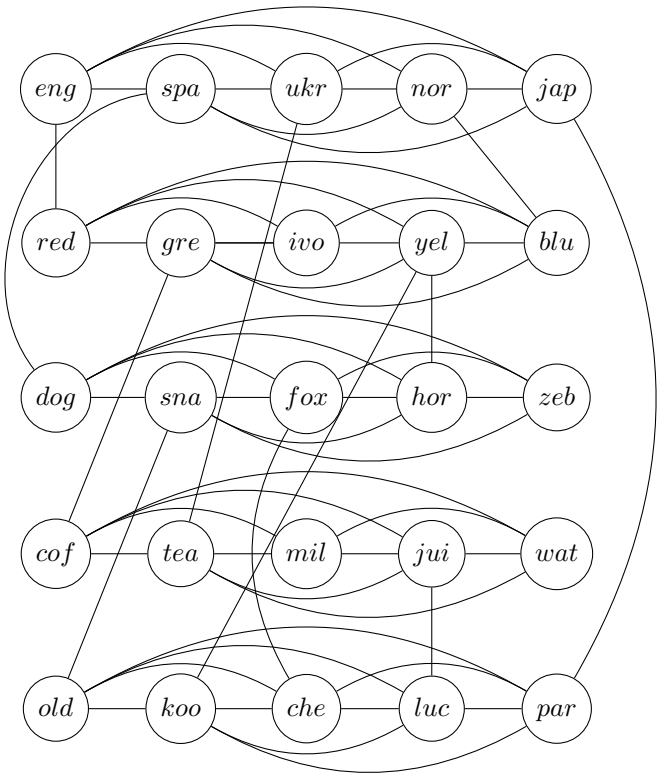

Figure 1: (3.1b) primal constraint graph of N

(c) No, N is not arc-consistent. We provide the arc-consistent constraint network $N' = \langle V, D', C \rangle$ with

$$D'_v = \begin{cases} \{1\}, & \text{if } v = norwegian \\ \{3\}, & \text{if } v = milk \\ \{2\}, & \text{if } v = blue \\ \{3,4\}, & \text{if } v = ivory \\ \{4,5\}, & \text{if } v \in \{coffee, green\} \\ \{2,3,4\}, & \text{if } v = horse \\ \{3,4,5\}, & \text{if } v \in \{english, red\} \\ \{2,4,5\}, & \text{if } v \in \{ukrainian, tea\} \\ \{2,3,4,5\}, & \text{if } v \in \{spanish, japanese, dog, parliament\} \\ \{1,2,4,5\}, & \text{if } v \in \{juice, water, lucky\} \\ \{1,3,4,5\}, & \text{if } v \in \{yellow, kool\} \\ \{1,2,3,4,5\}, & \text{otherwise} \end{cases}$$

That makes 32 domain values less. I've spent an hour and this, where my solver spent 1.5ms on this task. I'm glad we have the power to build machines for doing such tedious work instead of doing it by hand.

Exercise 4.2

(a) We show that $Sol(\mathcal{C}) = Sol(\mathcal{C}')$ by contradiction.

Let $\mathcal{C} = \langle V, D, C \rangle$ and $\mathcal{C}' = \langle V, D', C \rangle$. By definition each variable assignment $a(v_i)$ must be in its variable domain D_i, i.e., $a(v_i) \in D_i$. Therefore, a reduction of domains can only yield a network with less solutions. Since AC3 reduces domains, there can not be more solutions for $\mathcal{C}'$, i.e., $Sol(\mathcal{C}') \subseteq Sol(\mathcal{C})$.

Assume that solution $a \in Sol(\mathcal{C})$ is lost after the domain reduction by AC3, i.e., $a \notin Sol(\mathcal{C}')$. WLOG it follows that at some iteration of AC3 the value $a(v_i)$ was removed from the domain D_i', so it holds $a(v_i) \notin D_i'$. By the definition of the revise function of AC3, it holds that $D_i' \leftarrow D_i \cap \pi_i(R_{ij} \bowtie D_j)$. Since $a(v_i)$ is in D_i but not in D_i', it follows that $a(v_i) \notin \pi_i(R_{ij} \bowtie D_j) \implies (a(v_i), a(v_j)) \notin (R_{ij} \bowtie D_j)$.

The join $R_{ij} \bowtie D_j$ represents the set of all consistent assignments of variable v_j over the relation R_{ij}. With $(a(v_i), a(v_j)) \notin (R_{ij} \bowtie D_j)$ it follows that variable assignments $(a(v_i), a(v_j))$ are not consistent with the constraint over R_{ij}. It contradicts with our assumption that $a \in Sol(\mathcal{C})$, because a solution must satisfy all constraints of the network. We have shown that there is no solution $a \in Sol(\mathcal{C})$, such that $a \notin Sol(\mathcal{C}')$. □

(b) To show that AC3 produces an arc-consistent network $\mathcal{C}'$ we consider three cases. The first case is trivial, given an arc-consistent network the revise function does not remove any domain values and therefore does not modify the network.

Let $\mathcal{C}$ be a network with some not arc-consistent R_{ij}, i.e., there is some assignment a with $a(v_i) \in D_i$, $a(v_j) \in D_j$ and $(a(v_i), a(v_j)) \notin (R_{ij} \bowtie D_j)$. Initially, AC3 does revise all variable pairs (v_i, v_j) and (v_j, v_i) with (v_i, v_j) in some constraint's scope (R_{ij}). It follows that AC3 does revise (v_i, v_j) in respect of R_{ij} in some iteration and reduces $D_i' \leftarrow D_i \cap \pi_i(R_{ij} \bowtie D_j)$. Because $(a(v_i), a(v_j)) \notin (R_{ij} \bowtie D_j)$ implies $a(v_i) \notin \pi_i(R_{ij} \bowtie D_j)$, it holds that $a(v_i) \notin D_i'$. This reduces the initial arc-inconsitency of v_i relative to v_j.

By the domain value reduction, AC3 can introduce new not arc-consistent constraints. Let $a(v_i) \in D_i$ be a value, that was removed by the revise function for constraint R_{ij}. Since the revise function changed D_i, AC3 will again revise all variable pairs (v_k, v_i) with $k \neq i, k \neq j$ for all constraints R_{ki}. This removes all inconsistent values for all domains D_k,i.e., all $a(v_k) \in D_k$ with $a(v_k) \notin \pi_k(R_{ki} \bowtie D_i)$ are removed from D_k. All other constraints R_{lm} with $m \neq i$ are not directly affected by the reduced domain D_i and can therefore not become not arc-consistent by the revisal of (v_i, v_j).

We have shown, that all initial not arc-consistent constraints are made arc-consistent and the same holds for all newly introduced inconsistencies by the domain reduction. It follows that all remaining domains of $\mathcal{C}'$ form an arc-consistent network. □

(c) All trivial assignments with complexity of $\mathcal{O}(1)$ are left out of this analysis. Further it is assumed that set containment can be tested in $\mathcal{O}(1)$, while is is only true for fully expanded sets with exponential space requirements. We can, however, archieve $\mathcal{O}(1)^*$ using hash tables and amortized analysis in linear space. The natural tree-based approach yields $\mathcal{O}(\log n)$. For the queue, we assume a stack is used with $\mathcal{O}(1)$ insertion and removal (last inserted value). Let n be the number of variables, all domains have size $\leq k$ and e is the number of constraints.

The initialisation of the counters consists of an iteration over all constraints R_{ij} and values $a_i \in D_i$, $a_j \in D_j$, this takes $\Theta(e \cdot k^2)$ time.

Next we initialise the set $S[v_j, a_j]$ with the supported tuples, adjust the counters and move unsupported tuples into Q for each constraint R_{ij} and values $a_i \in D_i$, $a_j \in D_j$, again this takes $\Theta(e \cdot k^2)$ time. We notice that each set $S[v_j, a_j]$ and Q have at most $e \cdot k$ elements, also the counter value for any variable-value pair is $\leq e \cdot k$.

The while loop needs some special treatment based on the observation that $|S[v_j, a_j]| + |Q| = e \cdot k$ and others. But, due to limited time, I have to skip this part this time.